

AD-A102 569 CALIFORNIA UNIV BERKELEY CENTER FOR PURE AND APPLIED MATHEMATICS
A LOOK AHEAD LANCZOS ALGORITHM FOR UNSYMMETRIC MATRICES, (U)
JUN 81 B N PARLETT, D TAYLOR NO0014-76-C-0013
PAM-43 NL
UNCLASSIFIED

UNCLASSIFIED PAM-43

014-76-C-0013

NL

10E

688

102169

END
DATE
FILED
9-4-81

AD A102569

DTIC FILE COPY

PRO-0014-324
LEVEL

12

CENTER FOR PURE AND APPLIED MATHEMATICS
UNIVERSITY OF CALIFORNIA, BERKELEY

PAM-43

A LOOK AHEAD LANCZOS ALGORITHM FOR UNSYMMETRIC MATRICES

B. N. Parlett and D. Taylor

DTIC
SELECTED
AUG 2 1981
S D C

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

June 26, 1981

81 7 20 008

Legal Notice

This report was prepared as an account of work sponsored by the Center for Pure and Applied Mathematics. Neither the Center nor the Department of Mathematics, makes any warranty expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information or process disclosed.

(II) PAM-43

12

(6)

A LOOK AHEAD LANCZOS
ALGORITHM FOR UNSYMMETRIC
MATRICES

by

10

B.N./Parlett[†] and D./Taylor[‡]

(II) 26 JUN 74

12



[†] Department of Mathematics and Computer Sciences Division, Department of Electrical Engineering and Computer Science, University of California at Berkeley

[‡] Department of Mathematics, University of California at Berkeley

The first author gratefully acknowledges support by the Office of Naval Research Contract N00014-76-C-0013. The second author thanks the Remote Sensing Research program at the University of California, Berkeley, for use of its computer facility.

411750 11

Abstract. The two sided Lanczos algorithm is known to suffer from serious breakdowns. These occur when the associated moment matrix does not permit triangular factorization. We modify the algorithm slightly so that it corresponds to using a 2×2 "pivot" in triangular factorization whenever a 1×1 pivot would be dangerous. The incidence of breakdown is greatly reduced. The price paid is that the tridiagonal matrix produced by the algorithm now has bumps whenever a 2×2 pivot is used.

Accession For	
NTIS GFA&I	<input checked="" type="checkbox"/>
PTIC T&B	<input type="checkbox"/>
Unannounced	
Justification for filing	
On file	
Bv	
Distribution/	
Availability Codes	
Avail and/or	
Dist	Special
P	

1. THE LANCZOS ALGORITHM AND ITS BREAKDOWN

The most popular way to obtain all the eigenvalues of a nonsymmetric $n \times n$ matrix B is to use the QR algorithm which is readily available in most computing centers. As the order n increases above 100 the QR algorithm becomes less and less attractive, especially if only a few of the eigenvalues are wanted. This is where the Lanczos algorithm comes into the picture. It does not alter B at all but constructs a tridiagonal matrix J gradually by adding a row and column at each step. After several steps some of the eigenvalues θ_i of J will be close to some eigenvalues λ_k of B and by the n^{th} step, if nothing goes wrong, $\theta_i = \lambda_i$, $i = 1, \dots, n$. This description is correct in the context of exact arithmetic. Unfortunately things can go wrong, even in the absence of rounding errors. For the relation between these troubles and orthogonal polynomials see [2].

In order to discuss these troubles we must give some details about the algorithm. Let J_k be the $k \times k$ tridiagonal produced at step k of the algorithm. There are infinitely many tridiagonal matrices similar to B and J_n is one of them. Thus for some matrix $Q_n \equiv (q_1, \dots, q_n)$ we have

$$Q_n^{-1} B Q_n = J_n \quad (1)$$

It simplifies the exposition considerably to introduce a redundant symbol and write P_n^* instead of Q_n^{-1} . The superscript * indicates conjugate transpose.

Let $P_n \equiv (p_1, \dots, p_n)$ and replace (1) by two separate relations

$$P_n^* Q_n = I, \quad (2)$$

$$P_n^* B Q_n = J_n. \quad (3)$$

We mention in passing that when $B^* = B = A$ then we can arrange that $P_n = Q_n$. The difficulty we are going to describe cannot occur when $P_n \neq Q_n$.

By equating elements on each side of $BQ_n = Q_n J_n$ and $P_n^* B = J_n P_n^*$ in the natural increasing order we shall see that B , p_1 and q_1 essentially determine all the other elements of P_n , Q_n , and J_n . On writing

$$J_n \equiv \begin{bmatrix} \alpha_1 & \gamma_2 \\ \beta_2 & \alpha_2 & \gamma_3 \\ \beta_2 & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$$

we find

$$p_1^* B q_1 = \alpha_1 ,$$

and

$$B q_1 = q_1 \alpha_1 + q_2 \beta_2 , \quad p_1^* B = \alpha_1 p_1^* + \gamma_2 p_2^* .$$

Hence q_2 and p_2^* are, respectively, multiples of "residual" vectors

$$r_1 \equiv B q_1 - q_1 \alpha_1 , \quad s_1^* \equiv p_1^* B_1 - \alpha_1 p_1^* .$$

Furthermore, since $p_2^* q_2 = 1$ by (2), we have

$$s_1^* r_1 = \gamma_2 p_2^* q_2 \beta_2 = \gamma_2 \beta_2 \equiv \omega_2 .$$

If $\omega_2 \neq 0$ and β_2 is given any nonzero value then γ_2 , q_2 , and p_2^* are all determined uniquely. A good choice is $\beta_2 = \sqrt{1\omega_2}$.

The general pattern emerges at the next step, on equating the second columns on each side of $BQ_n = Q_n J_n$ and $P_n^* B = J_n P_n^*$,

$$p_2^* B q_2 = \alpha_2,$$

$$B q_2 = q_1 \gamma_2 + q_2 \alpha_2 + q_3 \beta_3, \quad p_2^* B = \beta_2 p_1^* + \alpha_2 p_2^* + \gamma_3 p_3^*.$$

At this point we can compute the "residual" vectors

$$r_2 \equiv B q_2 - q_1 \gamma_2 - q_2 \alpha_2, \quad s_2^* \equiv p_2^* B - \beta_2 p_1^* - \alpha_2 p_2^*$$

and

$$\omega_3 \equiv s_2^* r_2 = \gamma_3 \beta_3.$$

If $\omega_3 \neq 0$ and β_3 is given any nonzero value then γ_3 , q_3 , and p_3^* are all determined uniquely. And so it goes on until some ω_j vanishes.

This is the Lanczos algorithm. It must terminate at the n^{th} step with $\omega_{n+1} = 0$ but it may stop sooner.

Premature termination at say step $j (< n)$ can occur in two ways:

(I) either $r_j = 0$ or $s_j^* = 0^*$ or both,

or

(II) $r_j \neq 0$, $s_j^* \neq 0^*$, but $\omega_{j+1} = 0$.

In the 1950's when the Lanczos algorithm was regarded as a way to compute J_n Case I was regarded as a mild nuisance. If $r_j = 0$ then any nonzero vector orthogonal to p_1, \dots, p_j can be chosen as q_{j+1} . Similarly $s_j = 0$ gives ample choice for p_{j+1} .

Today, regarding Lanczos as a way to find a few eigenvalues of large B it seems better to stop at Case I in the knowledge that every eigenvalue of J_j is an eigenvalue of B . If more eigenvalues are wanted then it is best to start the Lanczos algorithm afresh with new, carefully chosen starting vectors q_1 and p_1 .

The real trouble, which cannot occur when $B = B^* = A$, is Case II. Wilkinson calls this a serious breakdown. There seems to be no choice but to start again but no one has been able to suggest a practical way to choose the new q_1 and p_1^* so as to avoid another wasted run of the algorithm. That is why the Lanczos method has not been used much when $B^* \neq B$. In this article we propose a modification of the algorithm which greatly reduces the occurrence of Case II. The price paid for this convenience is that J is not quite tridiagonal. There is a small bump (or bulge) in the tridiagonal form to mark each occurrence (or near occurrence) of Case II.

Example 1. (No breakdown)

$$B = \text{diag}(2, 3, 4), q_1^* = \frac{1}{2}(1, 1, 1), p_1^* = \frac{1}{2}(1, 2, 1).$$

Step 1: $\alpha_1 = 3, \omega_2 = \frac{1}{2}, \beta_2 = 1, \gamma_2 = \frac{1}{2},$

$$q_2^* = \frac{1}{2}(-1, 0, 1), p_2^* = (-1, 0, 1).$$

Step 2: $\alpha_2 = 3, \omega_3 = \frac{1}{2}, \beta_3 = 1, \gamma_3 = \frac{1}{2},$

$$q_3^* = \frac{1}{2}(1, -1, 1), p_3^* = \frac{1}{2}(1, -2, 1).$$

Step 3: $\alpha_3 = 3$, $\omega_4 = 0$.

$$J_3 = \begin{bmatrix} 3 & \frac{1}{2} & 0 \\ 1 & 3 & \frac{1}{2} \\ 0 & 1 & 3 \end{bmatrix}.$$

2. THE TWO SIDED GRAM-SCHMIDT PROCESS.

The serious breakdown described above is not limited to the Lanczos algorithm. It can occur in any attempt to use the familiar Gram-Schmidt process to produce a biorthogonal (or biorthonormal) pair of sequences. Our modification of Lanczos seems more natural in such a context.

Let $F = (f_1, \dots, f_n)$ and $G = (g_1, \dots, g_n)$ be given real nonsingular n by n matrices. In other words $\{f_1, \dots, f_n\}$ and $\{g_1, \dots, g_n\}$ are each a basis for the vector space \mathbb{R}^n of column n -vectors. We want to produce a new pair of bases $\{q_1, \dots, q_n\}$ and $\{p_1, \dots, p_n\}$ such that

$$P_n^* Q_n = \Omega_n = \text{diag}(\omega_1, \dots, \omega_n)$$

and, for each $j = 1, \dots, n$

$$\begin{aligned} \text{span } \{q_1, \dots, q_j\} &= \text{span } \{f_1, \dots, f_j\}, \\ \text{span } \{p_1, \dots, p_j\} &= \text{span } \{g_1, \dots, g_j\}. \end{aligned}$$

The Gram-Schmidt process dictates that at step j

$$q_j = f_j - \sum_{i=1}^{j-1} q_i (p_i^* f_j / \omega_i),$$

$$p_j = g_j - \sum_{i=1}^{j-1} p_i (p_i^* g_j / \omega_i),$$

$$\omega_j = p_j^* q_j.$$

All goes well until $\omega_j = 0$ for some j . This can happen despite the fact that F and G are nonsingular.

Example 2.

$$F = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}, G = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Step 1: $q_1 = f_1, p_1 = g_1, \omega_1 = 1$.

Step 2: $q_2^* = (0, 1, 0), p_2^* = (-1, 0, 1), \omega_2 = 0$.

One remedy for the case $\omega_j = 0$ is quite natural. If $p_j \neq 0$ then recompute q_j using f_{j+1} , in place of f_j . If this fails too then try f_{j+2} , and so on. If F is nonsingular there must be some $i > 0$ such that f_{j+i} will yield a nonzero value for ω_j .

Here is a formal proof of the last remarks. Let $q_j^{(k)}$ denote the vector obtained by using f_k instead of f_j , i.e.,

$$q_j^{(k)} = f_k - \sum_{i=1}^{j-1} q_i(p_i^* f_k / \omega_i).$$

Lemma: If $p_j \neq 0$ and $p_j^* q_j^{(k)} = 0$ for $k = j, j+1, \dots, n$ then F is singular.

Proof: By construction $p_j^* q_i = 0$ for $i < j$.

Hence $p_j \perp \text{span}(f_1, \dots, f_{j-1})$.

Thus $0 = p_j^* q_j^{(k)} = p_j^* f_k - 0$, for all $k \geq j$.

Thus $p_j \perp \text{span}(f_1, \dots, f_n)$.

Thus $p_j^* F = 0$ and F is singular. □

If the modification succeeds the first time then only one property of Gram-Schmidt has been sacrificed, namely

$$\text{span } (q_1, \dots, q_j) \neq \text{span } (f_1, \dots, f_j) .$$

Moreover, if no further breakdown occurs then

$$\text{span } (q_1, \dots, q_i) = \text{span } (f_1, \dots, f_i) \text{ for } i > j.$$

In many applications this price is well worth paying.

The Lanczos algorithm can be regarded as the two sided Gram-Schmidt process applied to the columns of the special matrices

$$K = K_n \equiv (q_1, Bq_1, B^2q_1, \dots, B^{n-1}q_1),$$

and

$$\tilde{K}^* = \tilde{K}_n^* \equiv (p_1^*, p_1^*B, p_1^*B^2, \dots, p_1^*B^{n-1}) .$$

We will not establish this result but content ourselves with stating the key observation, namely

$$\text{Span } (q_1, q_2, \dots, q_j, Bq_j) = \text{Span } (q_1, \dots, q_j, B^j q_1) .$$

The K matrices are called Krylov matrices and the pleasant fact is that most of the work required for general Gram-Schmidt disappears in this case because

$$p_i^* B q_j = 0 \text{ for } i < j-1 .$$

Thus the general formula

$$q_{j+1} = B^j q_1 - \sum_{i=1}^j q_i (p_i * B^j q_1 / \omega_i)$$

collapses to

$$q_{j+1} = B q_j - q_j \alpha_j - q_{j-1} \beta_{j-1}.$$

3. TRIANGULAR FACTORIZATION OF MOMENT MATRICES

Consider again the matrices K and \tilde{K}^* defined in the previous section. Note that the (i,j) element of $\tilde{K}^* K$ is $(p_1 * B^{i-1})(B^{j-1} q_1)$, so

$$\tilde{K}^* K = M = M(p_1, q_1); m_{i+1, j+1} = p_1 * B^{i+j} q_1.$$

In order to use this observation we need some basic facts about the Lanczos algorithm (see [4], [5], or [7]). If it does not breakdown it produces matrices P and Q such that

$$q_{j+1} = \chi_j(B) q_1 / (\prod_{i=2}^{j+1} \beta_i),$$

$$p_{j+1} = \chi_j(B^*) p_1 / (\prod_{i=2}^{j+1} \gamma_i),$$

where

$$\chi_{j+1}(t) = (t - \alpha_j) \chi_j(t) - \omega_j \chi_{j-1}(t),$$

and χ_j is the characteristic polynomial of the tridiagonal matrix J_j . In other words, for each j , q_j is a linear combination of the first j columns of K while p_j is the same linear combination of the columns of \tilde{K} , up to scaling. This can be expressed compactly in matrix

notation as

$$\begin{aligned} Q &= KL^{-*}\pi^{-1}, \\ P &= \tilde{K}L^{-*}\tilde{\pi}^{-1}. \end{aligned} \tag{4}$$

Here

$$\begin{aligned} \pi &= \text{diag}(1, \beta_2, \beta_2\beta_3, \dots), \\ \tilde{\pi} &= \text{diag}(1, \gamma_2, \gamma_2\gamma_3, \dots), \end{aligned}$$

and L is the unit lower triangular matrix such that $L^{-*} \equiv (L^{-1})^*$ has the coefficients of x_j above the diagonal in the j^{th} column.

Using (4) we can rewrite $I = P*Q$ as

$$I = P*Q = (\tilde{\pi}^{-1}L^{-1}\tilde{K}^*)(KL^{-*}\pi^{-1})$$

i.e.

$$M = L\Omega L^*, \tag{5}$$

where

$$\Omega = \tilde{\pi}\pi = \text{diag}(1, \omega_2, \omega_2\omega_3, \dots, \omega_2\dots\omega_n).$$

This result is not new (see Householder, [2]) but it is worth emphasizing that the product $\omega_2\dots\omega_i$ is the i^{th} pivot which arises in performing Gaussian elimination on the moment matrix M associated with B , q_1 and p_1 .

When $B \neq B^*$ the moment matrix M need not be positive definite and so triangular factorization need not be stable, even when M is nonsingular.

The best known remedy for instability is to use some form of row or column interchange whenever an ω_i is too small. The standard "partial pivoting" strategy is not available because a whole column of M is not known in the middle of the Lanczos process. An alternative remedy is to enlarge the notion of a "pivot" to include 2×2 , or even larger submatrices. This idea is discussed in Parlett and Bunch, 1971, [4]. It is the basis of our method. Whenever a 2×2 pivot is used the tridiagonal J bulges outwards temporarily.

In the context of the Lanczos process our remedy for a tiny ω_j requires us to compute q_{j+1} at the same time as q_j , and p_{j+1} at the same time as p_j . The formulas for these "Lanczos" vectors are somewhat different from the standard ones. We call our modification the "look-ahead Lanczos algorithm" because it computes at the current step some quantities which are not usually needed until the next step in the standard Lanczos process.

4. THE NEXT PIVOT

The decomposition $M = L \cdot L^*$ is never found explicitly. In order to make use of the idea of using a 2×2 pivot it is necessary to determine the top left 2×2 submatrix of what would be the reduced matrix in the triangular factorization of M . These three numbers can be determined from the information available in the Lanczos process.

After $i-1$ steps of the standard algorithm we have

$$r_i \equiv Bq_{i-1} - q_{i-1}\alpha_{i-1} - q_{i-2}\gamma_{i-1} = x_{i-1}(B)q_1/(\beta_2 \dots \beta_{i-1}),$$

$$s_i^* \equiv p_{i-1}^*B - \alpha_{i-1}p_{i-1}^* - \beta_{i-1}p_{i-2}^* = p_1^*x_{i-1}(B)/(\gamma_2 \dots \gamma_{i-1}),$$

$$\omega_i \equiv s_i^*r_i$$

Instead of normalizing r_i and s_i^* to get q_i and p_i^* we can look

ahead not to the next Lanczos vectors q_{i+1}, p_{i+1}^* but to any vectors r_{i+1}, s_{i+1}^* such that

$$\begin{aligned} \text{the plane } (r_i, r_{i+1}) &= \text{the plane } (q_i, q_{i+1}), \\ \text{the plane } (s_i^*, s_{i+1}^*) &= \text{the plane } (p_i^*, p_{i+1}^*). \end{aligned}$$

The simplest choice for r_{i+1} and s_{i+1}^* is

$$\begin{aligned} r_{i+1} &\equiv Br_i - q_{i-1}\omega_i, \\ s_{i+1}^* &\equiv s_i^*B - \omega_i p_{i-1}^*. \end{aligned}$$

The coefficient ω_i ensures that r_{i+1} is orthogonal to all the known p 's, namely p_1, \dots, p_{i-1} , and also that s_{i+1} is orthogonal to q_1, \dots, q_{i-1} .

Note that if we choose as q_i any vector in the plane (r_i, r_{i+1}) other than a multiple of r_i then q_i will be of the form

$$q_i = \psi(B)q_1 / (\beta_2 \dots \beta_i)$$

with ψ a monic polynomial of degree i instead of the usual x_{i-1} .

Moreover it will be possible to choose q_{i+1} to be of the same form, using a different ψ but of the same degree i . This is a mild generalization of the basic Lanczos algorithm. The degrees of the new Lanczos polynomials are still nondecreasing but they do not always go up by one at each step.

Before making such a choice we compute

$$\|r_i\|, \|s_i^*\|, \text{ and } \cos \angle (r_i, s_i^*) = \omega_i / \|r_i\| \|s_i^*\|.$$

Also

$$\omega_{i+1} = s_{i+1}^* r_{i+1}, \quad \theta_i = s_i^* r_{i+1} = s_{i+1}^* r_i, \\ \|r_{i+1}\|, \|s_{i+1}^*\|, \cos \angle(r_{i+1}, s_{i+1}) = |\omega_{i+1}| / \|r_{i+1}\| \|s_{i+1}^*\|.$$

Our choice of q_i , etc. must be based on the matrix

$$W_i = \begin{pmatrix} \omega_i & \theta_i \\ \theta_i & \omega_{i+1} \end{pmatrix}.$$

It turns out that the top left 2×2 submatrix of the reduced matrix in the associated triangular factorization of M is $(\omega_2 \dots \omega_{i-1}) W_i$, but no use will be made of this fact.

If both $\omega_i = 0$ and W_i is singular then more drastic measures are needed to salvage the algorithm. We will not pursue this case here. When $\omega_i = 0$ then the standard Lanczos algorithm breaks down. Yet if W_i is invertible it is easy to choose suitable q 's and p 's so that the algorithm can proceed.

The equations above can be condensed into block form.

$$(r_i, r_{i+1}) = B(q_{i-1}, r_i) - (q_{i-1}, r_i) \begin{pmatrix} \alpha_{i-1} & \omega_i \\ 0 & 0 \end{pmatrix} - q_{i-2}(r_{i-1} 0),$$

$$(s_i, s_{i+1})^* = (p_{i-1}, s_i)^* B - \begin{pmatrix} \alpha_{i-1} & 0 \\ \omega_i & 0 \end{pmatrix} (p_{i-1}, s_i)^* - \begin{pmatrix} \beta_{i-1} \\ 0 \end{pmatrix} p_{i-2}^*,$$

$$W_i = (s_i, s_{i+1})^* (r_i, r_{i+1}) \stackrel{\text{def}}{=} \begin{pmatrix} \omega_i & \theta_i \\ \theta_i & \omega_{i+1} \end{pmatrix}.$$

Various factorizations of W_i yield various selections for new q 's and p 's. These are discussed below. We write $\hat{\omega}$ for ω_{i+1} and $\hat{\theta}$ for θ_i .

Let us drop the subscript i and write

$$W = VU.$$

1. LU factorization

$$V = \begin{pmatrix} 1 & 0 \\ \theta/\omega & 1 \end{pmatrix}, \quad U = \begin{pmatrix} \omega & \theta \\ 0 & \hat{\omega} - \theta^2/\omega \end{pmatrix}$$

2. UL factorization

$$V = \begin{pmatrix} \omega - \theta^2/\omega & \theta \\ 0 & \hat{\omega} \end{pmatrix}, \quad U = \begin{pmatrix} 1 & 0 \\ \theta/\hat{\omega} & 1 \end{pmatrix}$$

3. QR factorization

$$V = \begin{pmatrix} \omega & -\theta \\ \theta & \omega \end{pmatrix} \tau^{-1}, \quad U = \begin{pmatrix} \tau^2 & \theta(\omega + \hat{\omega}) \\ 0 & (\omega\hat{\omega} - \theta^2) \end{pmatrix} \tau^{-1}, \quad \tau = (\omega^2 + \theta^2)^{1/2}$$

4. LU with interchange

$$V = \begin{pmatrix} \omega/\theta & 1 \\ 1 & 0 \end{pmatrix}, \quad U = \begin{pmatrix} \theta & \hat{\omega} \\ 0 & \theta - \omega\hat{\omega}/\theta \end{pmatrix}$$

5. THE SMALLEST ANGLE

In order to determine the smallest angle between the planes (r_i, r_{i+1}) and (s_i, s_{i+1}) it is best to orthonormalize the bases. Let the result be

$$(\bar{r}_i, \bar{f}_{i+1}) \text{ and } (\bar{s}_i, \bar{g}_{i+1}),$$

when the Gram-Schmidt process is used. The (matrix) angle between the two planes, call it ϕ , is defined, see [1], by

$$\cos \phi = (\bar{s}_i, \bar{g}_{i+1})^* (\bar{r}_i, \bar{f}_{i+1}).$$

Let the Singular Value Decomposition of $\cos \phi$ be

$$\cos \phi = U \Sigma V^*$$

where $\Sigma = \text{diag}(\sigma_1, \sigma_2)$ and $\sigma_1 \geq \sigma_2$. The appropriate definition of new Lanczos vectors to ensure the smallest $\angle(q_i, p_i)$ is

$$(q_i, q_{i+1}) = (\bar{r}_i, \bar{f}_{i+1}) V \Sigma^{-1/2},$$
$$(p_i, p_{i+1}) = (\bar{s}_i, \bar{g}_{i+1}) U^* \Sigma^{-1/2}.$$

Comments. No. 1 corresponds to the standard Lanczos algorithm. No. 2 corresponds to simply swapping s_i^* with $s_i^* B$ and r_i with $B r_i$ in the Lanczos algorithm. One consequence is that the J matrix will bulge out of tridiagonal form on both sides. No. 3 is always stable and keeps the bulge on one side. The same advantage accrues from No. 4 and, as might be expected, is somewhat simpler than No. 3.

We want to use No. 1 whenever this is a reasonable strategy but when ω is zero or very small it is vital to choose one of the other procedures. We have tentatively chosen No. 4 on the grounds of simplicity. The interesting question which we now address is when to use No. 4.

Initial success with No. 4 has not encouraged us to implement No. 5.

Criterion for Choosing a 2×2 Pivot

When Factorization No. 4 is chosen then

$$(q_i, q_{i+1}) = (r_i, r_{i+1}) \begin{pmatrix} 1 & -\omega_{i+1}/\theta_i \\ 0 & 1 \end{pmatrix},$$

$$(p_i, p_{i+1}) = (s_{i+1}, s_i) \begin{pmatrix} 1 & -\omega_i/\theta_i \\ 0 & 1 \end{pmatrix},$$

to within scalar multiples. Note the interchange. Hence

$$\cos \angle (q_i, p_i) = \cos \angle (r_i, s_{i+1}) = \frac{\theta_i}{r_i \cdot s_{i+1}} \equiv \psi_i$$

$$\cos \angle (q_{i+1}, p_{i+1}) = \frac{(\theta_i - \omega_i \omega_{i+1}/\theta_i)}{\|r_{i+1} - (\frac{\omega_{i+1}}{\theta_i}) r_i\| \cdot \|s_i - (\frac{\omega_i}{\theta_i}) s_{i+1}\|} = \psi_{i+1}.$$

Both these numbers are readily computable, without forming the new vectors, provided that $r_{i+1}^* r_i$ and $s_{i+1}^* s_i$ are known. The choice between No. 1 and No. 4 reduces to a comparison of

$$\phi_1 = |\cos \angle(r_i, s_i)| \text{ and } \phi_2 = \min \{|\psi_i|, |\psi_{i+1}|\}.$$

If $\phi_1 < 100\epsilon$ and $\phi_2 < 100\epsilon$ then our algorithm stops and reports failure. Otherwise, for a given bias factor we proceed as follows: if $\phi_1 \geq (\text{bias factor}) \phi_2$ then use standard Lanczos else use Factorization No. 4. When bias = 0 we recover the standard algorithm. Currently bias = 1.

Sometimes the test declares that a standard Lanczos step is safe. In such cases ψ_1 and ψ_2 are not used and their computation may be regarded as an overhead of 4 inner products. No matrix-vector products are "wasted".

The 4 inner products arise as follows. We need

$$\|\tilde{s}_i^*\|^2 = \|s_i^* - \left(\frac{\omega_i}{\theta_i}\right)s_{i+1}^*\|^2 = \|s_i\|^2 - 2\left(\frac{\omega_i}{\theta_i}\right)s_i^*s_{i+1} + \left(\frac{\omega_i}{\theta_i}\|s_{i+1}\|\right)^2$$

$$\|\tilde{r}_{i+1}\|^2 = \|r_{i+1} - \left(\frac{\omega_{i+1}}{\theta_i}\right)r_i\|^2 = \|r_{i+1}\|^2 - 2\left(\frac{\omega_{i+1}}{\theta_i}\right)r_i^*r_{i+1} + \left(\frac{\omega_{i+1}}{\theta_i}\|r_i\|\right)^2.$$

We may regard the second and third terms on the right hand sides as the price to be paid for knowing that a standard Lanczos step is safe. Observe that \tilde{r}_{i+1} and \tilde{s}_i^* are not computed.

Let us write $R_i = (r_i, r_{i+1})$, $S_i = (s_i, s_{i+1})$.

Then the look-ahead part of the algorithm comprises the computation of r_{i+1} , s_{i+1} and the unknown elements of $S_i^*R_i$, $R_i^*R_i$, and $S_i^*S_i$. Before specifying the algorithm we describe the bumpy tridiagonal matrix J .

5. J , the Projection of B

The standard (biorthogonal) Lanczos algorithm produced a tridiagonal matrix J_j by the end of step j . The look-ahead algorithm produced a block tridiagonal matrix, also called J_j , and written as

$$J_j = \begin{bmatrix} A_1 & \Gamma_2 & & \\ B_2 & A_2 & \Gamma_3 & \\ & B_3 & A_3 & \cdot \\ & \cdot & \cdot & \Gamma_j \\ & B_j & A_j & \end{bmatrix}$$

The diagonal blocks are capital α 's and the subdiagonal blocks are capital β 's. The A_i are either 1 by 1 or 2 by 2 and the B_i and T_i are shaped conformably. For convenience in finding eigenvalues of J we have forced each B_i to have one of the following four forms

$$[+] , [0+] , \begin{bmatrix} + \\ 0 \end{bmatrix} , \begin{bmatrix} 0 & + \\ 0 & 0 \end{bmatrix},$$

where + stands for a positive quantity. It turns out that each T_i also has rank one.

The left and right Lanczos vectors will be grouped by step and written as $P_1^*, P_2^*, \dots, P_j^*$ and Q_1, \dots, Q_j where sometimes Q_i is n by 1 and sometimes n by 2. We collect the vectors together in $\hat{Q} = (Q_1, \dots, Q_j)$ and $\hat{P}_j = (P_1, \dots, P_j)$. The matrix $\hat{Q}_j \hat{P}_j^*$ is the projector matrix onto the left and right Krylov subspaces and B 's (oblique) projection into them is defined by

$$(\hat{Q}_j \hat{P}_j^*) B (\hat{Q}_j \hat{P}_j^*) = \hat{Q}_j J_j \hat{P}_j^*.$$

Thus J_j is the representation of this projection with respect to the bases given by the rows of \hat{P}_j^* and the columns of \hat{Q}_j .

Please note that j is not the order of J_j .

6. The Look-ahead Lanczos Algorithm (called LAL hereafter)

We have chosen our notation to camouflage the complications caused by the fact that each step may be either a single one or a double one. It turns out that quantities are computed in a somewhat different order and way from the standard two sided Lanczos algorithm (called LAN) and the reader may

find the differences loom larger than the similarities. We have found it helpful to think that step i takes certain residual matrices R_i and S_i , decides how to modify them, then turns them into the new Q_i and P_i^* , and finally computes part of the next set of residual matrices.

It is convenient to define the index ℓ by

$$\ell = 1 + \text{order } (\hat{Q}_{i-1}).$$

Thus by the end of step i we shall have

$$Q_i = \begin{cases} q_\ell & , \text{ if step } i \text{ is single,} \\ (q_\ell, q_{\ell+1}) & , \text{ if step is double.} \end{cases}$$

Similarly for P_i^* . However, in all cases, $R_i = (r_\ell, r_{\ell+1})$ and $S_i = (s_\ell, s_{\ell+1})$.

In describing LAL we call any computation involving n scalar multiplications or divisions a vector operation and abbreviate it as 1 v.op.. The algorithm requires that the user supply a subroutine (or subroutines) for computing Bx and y^*B from x and y^* . The cost of these operations will be problem dependent. We stress that this is the only way in which B enters the process.

Step i of LAL

On hand are P_{i-1}^* , Q_{i-1} (both are null when $i = 1$), and z_i which is a multiple of column 1 of Γ_i ($z_1 = 1$), together with non null residual vectors r_ℓ , s_ℓ^* and scalars $w = w_\ell = s_\ell^* r_\ell$, $\|r_\ell\|$, $\|s_\ell^*\|$.

1. Look-Ahead

(a) Complete $R_i = (r_\ell, r_{\ell+1})$ and $S_i^* = (s_\ell, s_{\ell+1})^*$.

$$r_{\ell+1} = Br_\ell - Q_{i-1}z_i\omega,$$

$$s_{\ell+1}^* = s_\ell^*B - \omega p_{\ell-1}^*$$

(2 matrix-vector products + 2 or 3 vector ops.)

(b) Compute needed inner products.

$$\theta = s_\ell^*r_{\ell+1} = s_{\ell+1}^*r_\ell, \hat{\omega} = s_{\ell+1}^*r_{\ell+1},$$

$$r_\ell^*r_{\ell+1}, s_\ell^*s_{\ell+1}, \|r_{\ell+1}\|, \|s_{\ell+1}\|.$$

(6 v. ops.)

(c) Compute cosines of important angles.

$$\phi_1 = \cos \angle(r_\ell, s_\ell) = \omega / \|r_\ell\| \cdot \|s_\ell\|, \phi_2 = 0,$$

if $\theta = 0$ then go to step 2, otherwise

$$\tau_1 = \omega / \theta, \tau_2 = \hat{\omega} / \theta;$$

$$\|\tilde{r}_{\ell+1}\| = \sqrt{(\|r_{\ell+1}\|^2 - 2\tau_2(r_\ell^*r_{\ell+1}) + \tau_2^2\|r_\ell\|^2)}$$

$$\|\tilde{s}_\ell\| = \sqrt{(\|s_\ell\|^2 - 2\tau_1(s_\ell^*s_{\ell+1}) + \tau_1^2\|s_{\ell+1}\|^2)}$$

$$\psi_1 = \cos \angle(r_\ell, s_{\ell+1}^*) = \theta / \|r_\ell\| \cdot \|s_{\ell+1}^*\|,$$

$$\psi_2 = \cos \angle(\tilde{r}_{\ell+1}, \tilde{s}_\ell^*) = (\omega\tau_2 - \theta) / \|\tilde{r}_{\ell+1}\| \cdot \|\tilde{s}_\ell^*\|,$$

$$\phi_2 = \min \{|\psi_1|, |\psi_2|\}.$$

(0 v.ops.)

2. Test for failure: if $|\phi_1| < \text{tol}$ and $\phi_2 < \text{tol}$ then exit with error message.

3. Select: if $|\phi_1| > (\text{bias factor}) \cdot \phi_2$ then take a single step, otherwise take a double step.

4. Single

Double

(a) factor $W = \begin{pmatrix} \omega & \theta \\ \theta & \hat{\omega} \end{pmatrix} = VU$.

$$\beta_\ell = \|r_\ell\| \sqrt{\phi_1}$$

$$\Delta = \text{diag } (\beta_\ell, \beta_\ell \beta_{\ell+1})$$

$$\gamma_\ell = \omega / \beta_\ell$$

$$= \text{diag } (\|r_\ell\| \sqrt{\phi_1}, \|r_{\ell+1}\| \sqrt{\phi_2})$$

(0 v. ops.)

(0 v. ops.)

$$V = \begin{pmatrix} \tau_1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \hat{\theta} & 0 \\ 0 & \hat{\theta} - \omega \tau_2 \end{pmatrix} \Delta^{-1}$$

$$U = \Delta \begin{pmatrix} 1 & \tau_2 \\ 0 & 1 \end{pmatrix}$$

(b) form Q_i and P_i^*

$$q_\ell = r_\ell / \beta_\ell ,$$

$$Q_i = R_i U^{-1} ,$$

$$p_\ell^* = s_\ell^* / \gamma_\ell ,$$

$$P_i^* = V^{-1} S_i^* ,$$

(2 v. ops.)

(6 v. ops.)

Single

(c) complete Γ_i , B_i

$$\Gamma_i = z_i \gamma_\ell$$

$$B_i = \beta_\ell \text{ or } (0 \ \beta_\ell)$$

(0 v. ops.)

Double

$$\Gamma_i = z_i (\omega, \psi_2)^{-1}$$

$$B_i = \begin{pmatrix} \beta_\ell \\ 0 \end{pmatrix} \text{ or } \begin{pmatrix} 0 & \beta_\ell \\ 0 & 0 \end{pmatrix}$$

(0 v. ops.)

(d) form new residuals

$$r_{\ell+1} = r_{\ell+1} / \beta_\ell,$$

$$s_{\ell+1}^* = s_{\ell+1}^* / \gamma_\ell,$$

(2 v. ops.)

$$r_{\ell+1} = (BQ_i - Q_{i-1}\Gamma_i) \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

$$s_{\ell+2}^* = p_\ell^* B - \beta_\ell p_{\ell-1}^*$$

$$= (0 \ 1) (P_i^* B - B_i P_{i-1}^*),$$

(2 matrix-vector products

+ 2 or 3 v. ops.)

(e) form A_i

$$\alpha_\ell = 1/\tau_1$$

(0 v. ops.)

$$A_i = \begin{bmatrix} \tau_2 & p_\ell^* B q_{\ell+1} \\ \beta_{\ell+1} & -\frac{\tau_1 \beta_{\ell+1} \psi_1}{\psi_2} & p_\ell^* B q_{\ell+1} \end{bmatrix}$$

(1 v. op.)

Single

Double

(f) orthogonalize

$$r_{\ell+1} \leftarrow r_{\ell+1} - q_\ell \alpha_\ell ,$$

$$s_{\ell+1}^* \leftarrow s_{\ell+1}^* - \alpha_\ell p_\ell^* ,$$

(2 v. ops.)

$$r_{\ell+1} \leftarrow r_{\ell+2} - q_i A_i \begin{pmatrix} 0 \\ 1 \end{pmatrix} ,$$

$$s_{\ell+2}^* \leftarrow s_{\ell+2}^* - (1 \ 0) A_i P_i^* ,$$

(4 v. ops.)

(g) inner products for next step

$$\| r_{\ell+1} \| \leftarrow (\| r_{\ell+1} \|^2 - 2\alpha_\ell r_\ell^* r_{\ell+1} + \alpha_\ell^2 \| r_\ell \|^2)^{1/2} / \beta_\ell \quad \text{compute}$$

$$\| s_{\ell+1} \| \leftarrow (\| s_{\ell+1} \|^2 - 2\alpha_\ell s_\ell^* s_{\ell+1} + \alpha_\ell^2 \| s_\ell \|^2)^{1/2} / \gamma_\ell \quad \| r_{\ell+2} \| ,$$

$$\omega_{\ell+1} \leftarrow \hat{\omega} / \omega - \omega_\ell^2$$

(0 v. ops.)

$$\| s_{\ell+2} \| ,$$

$$\omega_{\ell+2} = s_{\ell+2}^* r_{\ell+2}$$

(3 v. ops.)

(h) reset z,

$$z_{i+1} = (1)$$

$$z_{i+1} = \begin{pmatrix} 1 \\ -\omega \cdot w_2 / (\beta_\ell^2 \beta_{\ell+1}) \end{pmatrix}$$

end of step i of LAL.

Total Cost of step i .

Look-ahead: 2 matrix-vector prods + 9 v. ops.

Single step: 6 v. ops.

Double step: 2 matrix-vector prods + 16 v. ops.

For comparison we note that the standard two sided Lanczos algorithm which keeps $\| p_\lambda^* \| = \| q_\lambda \|$ requires 2 matrix-vectors prods + 10 v. ops.

There are 3 different ways of advancing two steps

LAN	:	4 matrix-vector p's + 20 v. ops.,
LAL, 1 double step :	"	+ 25 v. ops.,
LAL, 2 single steps:	"	+ 30 v. ops.

The bias factor in Step 3 is a programming device which permits LAL to implement standard Lanczos ($bias = 0$) or a sequence of double steps ($bias = \infty$). We do not claim that our setting ($bias = 1$) is optimal, but we doubt that it is far off.

7. Numerical Examples

We give a few examples which contrast the performance of the standard two sided Lanczos algorithm and our lookahead variant. They illustrate the stabilizing effect of the new variant.

As mentioned earlier there are several aspects of the problem which we have not yet addressed. The most important is the efficient computation of eigenvalues of J . All our computations were done on the NOVA 840 of the Remote Sensing Research Program at the University of California, Berkeley.

This forced us to use fairly small examples and this permitted us to use the QR programs from EISPACK to determine J's eigenvalues.

The Look-ahead algorithm reduces to the regular algorithm when the bias factor = 0 . We kept the bias = 1 for all our examples.

Explanation of the Tables

Each table gives snapshots of a Lanczos run, exhibiting what we feel is the essential information.

ℓ - the order of the J-matrix.

ϕ_1, ϕ_2 - cosines of angles between various candidates for p_ℓ^* and q_ℓ . See section 4 for precise definition.

$\rho(J)$ - the spectral radius of J .

The goal of our algorithm is to keep ϕ_1 from sudden plunges towards 0 . We could have used $\|J\|$ instead of $\rho(J)$, as an indication of "stability". We fear the appearance of arbitrarily large "spurious" eigenvalues in J . We expect some of J's eigenvalues to stabilize, as ℓ increases, at certain points in the complex plane. These points should be part of B's spectrum.

If a double step occurs in the Lookahead algorithm for a particular value of ℓ then ϕ_1 and ϕ_2 are not defined at $\ell + 1$ and such places are indicated by dashes.

Example I:

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$r_1 = s_1 = [1,2,3,4,5,6]^* \quad \text{no. of steps} = 6$$

The eigenvalues of B are the sixth roots of unity. The size of the matrix allows for the complete history.

TABLE I

ℓ	Reg. Lanczos			Look-ahead Lanczos		
	ϕ_1	ϕ_2	$\rho(J)$	ϕ_1	ϕ_2	$\rho(J)$
1	1.000	.1277	.8351	1.000	.1277	.8351
2	.1281	.0077	1.503	.1281	.0077	1.503
3	.0072	10^{-6}	1.012	.0072	10^{-6}	1.012
4	10^{-6}	.0488	10^{+6}	10^{-6}	.0488	----
5	10^{-7}	10^{-7}	----	----	----	4.781
6	----	----	----	.0068	.0068	----
7	----	----	----	----	----	1.000

Comment:

Steps 1 - 3 of both Regular and Look-ahead Lanczos are identical. At step 4, Regular Lanczos normalizes s_4 and r_4 by factors of 10^{-3} , producing elements in J of size 10^6 . Step 5 of Regular Lanczos is then aborted because the vectors are now orthogonal to working precision.

The Look-ahead algorithm avoids the large element growth in J , by doing a double step. The resulting J -matrix had eigenvalues identical to B to working accuracy.

II. B is 12×12 block upper triangular with diagonal blocks shown below.

The upper elements b_{ij} were randomly distributed in $[-5, 5]$.

$$B_1 = \begin{bmatrix} 95 & 2 \\ -2 & 95 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 2 & -95 \\ 95 & 2 \end{bmatrix}, \quad B_3 = 99, \quad B_4 = B_5 = 98,$$

$$B_6 = \begin{bmatrix} 25 & -50 \\ 50 & 25 \end{bmatrix}, \quad B_7 = 10^{-2}, \quad B_8 = \begin{bmatrix} 50 & -50 \\ 50 & 50 \end{bmatrix}$$

$$r_1 = s_1 = [1, \dots, 1]^*$$

No. of steps taken = 24.

TABLE II

	Reg. Lanczos			Look-ahead Lanczos		
λ	ϕ_1	ϕ_2	$\sigma(J)$	ϕ_1	ϕ_2	$\sigma(J)$
2	.2075	.6644	269.3	.2075	.6644	-----
3	.2101	.2059	83.95	-----	-----	83.95
4	.7592	.4422	108.5	.7594	.4421	108.5
5	.4405	.4718	96.67	.4403	.4717	-----
6	.3438	.1787	98.45	-----	-----	98.45
7	.1304	.0914	98.40	.1304	.0914	98.40
8	.1217	.0641	98.57	.1217	.0651	98.57
19	.0012	.0389	10^{10}	.1386	.0373	107.2
20	.0012	.0389	10^{12}	.2115	.2661	-----
21	.0012	.0389	10^{13}	-----	-----	164.5
22	.0012	.0389	10^{11}	.1707	.0109	122.0
23	.0012	.0389	10^{14}	.0164	.0706	-----
24	.0012	.0389	10^{18}	-----	-----	164.5

Comment on Example II.

B is fairly far from normal. After 24 steps, 7 of J's eigenvalues had stabilized to between 3 and 7 decimal places using the Look-ahead algorithm. What is surprising is that the regular Lanczos algorithm produced J's for which 6 eigenvalues had stabilized to 3 or more decimal places despite elements in J as large as 10^{18} .

III. B is a 15 by 15 upper triangular matrix with evenly spaced real eigenvalues. The super diagonal elements were random in the range of [-10, 10]. The diagonal elements were

$$b_{jj} = \begin{cases} 10j - 75, & j = 1, \dots, 7, \\ 0, & j = 8, \\ 10j - 85, & j = 9, \dots, 15. \end{cases}$$

$$r_1 = s_1 = (1, 1, \dots, 1)^*.$$

TABLE III.

k	Reg. Lanczos			Look-ahead Lanczos		
	ϕ_1	ϕ_2	$\rho(J)$	ϕ_1	ϕ_2	$\rho(J)$
7	.3205	.0580	64.87	.3205	.0580	64.87
8	.0646	.2167	154.2	.0646	.2167	-----
9	.0539	.0535	65.02	-----	-----	65.02
10	.2843	.2336	65.00	.2839	.2353	65.00
11	.1651	.2007	65.00	.1659	.1990	-----
12	.1516	.0447	65.00	-----	-----	65.00
13	.0443	.1934	134.2	.1439	.1563	-----
14	.0391	.0303	160.0	-----	-----	65.00
15	.0183	.0262	201.8	.3650	.1736	65.00
19	.0438	.0016	249.1	.2426	.4273	-----
20	.0070	.0896	676.2	-----	-----	65.00
21	.0072	.0068	281.8	.1691	.2296	-----
22	.0368	.0176	279.0	-----	-----	71.40
23	.0269	.0306	290.6	.0704	.0284	65.00
30	.0303	.1304	332.6	.0518	.0114	111.7

Comment: The run was halted after 30 steps.

LAL: To 14 eigenvalues of B there were approximations good to between 4 and 7 decimals, including duplicates of the two extremes.

LAN: To 4 eigenvalues of B there were approximations good to between 4 and 7 decimals.

IV. 100 x 100 diagonal matrix

$\text{diag } (B) = (1, 2, \dots, 20, 41, 62, \dots, 440, 481, 522, \dots, 1260, 1321, 1382, \dots, 2480, 2561, 2642, \dots, 4100).$

r_1, s_1^* random but unequal

TABLE IV.

l	Reg. Lanczos			Look-ahead Lanczos		
	ϕ_1	ϕ_2	$\rho(J)$	ϕ_1	ϕ_2	$\rho(J)$
4	.0633	.0037	3546.	.0633	.0037	3546.
5	.0036	.0163	10^4	.0036	.0163	-----
6	.0035	.0035	4648.	-----	-----	4669.
7	.0541	.0493	4489.	.0521	.0460	4303.
8	.0536	.0187	4126.	.0626	.0444	4219.
9	.0806	.0625	4866	.0519	.0027	4129.
10	.0770	.0582	10^4	.0029	.0187	-----
11	.0823	.0782	10^4	-----	-----	4082.
12	.0815	.0801	10^5	.0052	.0158	-----
13	.0821	.0819	10^5	-----	-----	4180.
14	.0819	.0818	10^5	.0478	.0367	4180.
15	.0820	.0820	10^6	.0425	.0204	4099.
16	.0820	.0820	10^6	.0225	.0091	4100.
17	.0820	.0820	10^7	.0156	.0042	4102.
18	.0820	.0820	10^7	.0089	.0098	-----
19	.0820	.0820	10^8	-----	-----	4100.

Comment:

After 19 steps 4 eigenvalues of B were represented in J to 3 or more decimal places. The standard Lanczos lost stability and no eigenvalues were represented in its J .

8. Conclusion

The Look-ahead algorithm is more complicated than the standard two-sided Lanczos process. In return it seems to prevent the serious breakdown which afflicts the standard program. However, much more testing is needed.

Other research topics which are relevant to a satisfactory solution of the eigenvalue problem for large nonnormal matrices are:

- (i) the use of reorthogonalization, or selective orthogonalization.
to maintain bi-orthogonality.
- (ii) the use of the Hyman-Laguerre method for finding eigenvalues of J .
- (iii) the use of the LR algorithm with interchanges for finding eigenvalues of J .
- (iv) incorporation of residual error bounds as termination criteria.

Saad studies alternative techniques which produce banded Hessenberg J -matrices in [8] and he gives a theoretical treatment of the standard two-sided Lanczos algorithm in [9].

References

- [1]. C. Davis and W. Kahan, "The Rotation of Eigenvectors by a Perturbation-III," SIAM J. on Num. Anal. 7 (1970), 1-46.
- [2]. W. Gragg, Notes from a "Kentucky Workshop" on the moment problem and indefinite metrics.
- [3]. W. Kahan, B. N. Parlett, and E. Jiang, "Residual Bounds on Approximate Eigensystems of Nonnormal matrices," to appear in SIAM J. on Num. Anal.
- [4]. A. S. Householder, The Theory of Matrices in Numerical Analysis, (Blaisdell Publ. Co., 1964).
- [5]. C. Lanczos, "An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators," Jour. Res. Nat. Bur. Stands. vol. 45 (1950), 255-282. See pp. 266-270.
- [6]. B. N. Parlett and J. R. Bunch, "Direct Methods for Solving Symmetric Indefinite Systems of Linear Equations," SIAM J. on Num. Anal. 8 (1971) 639-655.
- [7]. J. H. Wilkinson, The Algebraic Eigenvalue Problem, (Oxford Univ. Press, 1965).
- [8]. Y. Saad, "Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices," Lin. Alg. App. 34 (1980) 269-295.
- [9]. Y. Saad, "The Lanczos biorthogonalization algorithm and other oblique projection methods for solving large unsymmetric eigenproblems," to appear in SIAM Jour. Num. Anal. in 1982.

This report was done with support from the Center for Pure and Applied Mathematics. Any conclusions or opinions expressed in this report represent solely those of the author(s) and not necessarily those of The Regents of the University of California, the Center for Pure and Applied Mathematics or the Department of Mathematics.

